A Very High Speed True Random Number Generator with Entropy Assessment

Abdelkarim Cherkaoui, Viktor Fischer, Laurent Fesquet, Alain Aubert

A Very High Speed True Random Number Generator with Entropy Assessment

Abdelkarim Cherkaoui[1,2], Viktor Fischer[1], Laurent Fesquet[2], and Alain Aubert[1]

[1] Hubert Curien Laboratory, UMR CNRS 5516, Saint-Etienne, France
{abdelkarim.cherkaoui, fischer, alain.aubert@univ-st-etienne.fr}
[2] TIMA Laboratory, UMR CRNS 5159, Grenoble, France
{Laurent.Fesquet@imag.fr}

Abstract. The proposed true random number generator (TRNG) exploits the jitter of events propagating in a self-timed ring (STR) to generate random bit sequences at a very high bit rate. It takes advantage of a special feature of STRs that allows the time elapsed between successive events to be set as short as needed, even in the order of picoseconds. If the time interval between the events is set in concordance with the clock jitter magnitude, a simple entropy extraction scheme can be applied to generate random numbers. The proposed STR-based TRNG (STRNG) follows AIS31 recommendations: by using the proposed stochastic model, designers can compute a lower entropy bound as a function of the STR characteristics (number of stages, oscillation period and jitter magnitude). Using the resulting entropy assessment, they can then set the compression rate in the arithmetic post-processing block to reach the required security level determined by the entropy per output bit. Implementation of the generator in two FPGA families confirmed its feasibility in digital technologies and also confirmed it can provide high quality random bit sequences that pass the statistical tests required by AIS31 at rates as high as 200 Mbit/s.

Keywords: Random number generators, Self-timed rings, Stochastic models, Cryptography engineering

1 Introduction

Random number generators (RNGs) are crucial in cryptographic systems. They are used to generate confidential keys, challenges, and padding values, they are also used in authentication protocols and even in countermeasures against hardware attacks. Two kinds of generators and their combinations exist: pseudo-random and true random number generators (PRNGs and TRNGs, respectively). PRNGs are usually faster and their outputs have better statistical properties, but the numbers generated are predictable. TRNGs mostly exploit certain analog physical processes as a source of randomness. They are usually much slower and give statistically weaker results. However, they are preferred in applications

with high security requirements because their output is unpredictable. Besides unpredictability, good TRNGs must also fulfill another security requirement: they must not be manipulable [1].

According to new AIS31 evaluation criteria [2], unpredictability should be verified using a stochastic model to estimate entropy per bit. If entropy per output bit converges to one, the generator can be considered as unpredictable. Concerning the robustness of the generator against manipulations and environmental fluctuations, there are two possible solutions plus their combination: the generator can use a source of randomness that is not manipulable (e.g. thermal noise) and/or the source of randomness can be continuously tested.

Although security is the main requirement in cryptographic applications, to date very few published TRNG designs have been thoroughly evaluated from this point of view. For some designs such as [3], [4], the stochastic models are not feasible or at least not plausible, because they combine intrinsically pseudo randomness with true randomness. For other designs such as [5], [6], the model should be feasible, but this has not been suggested up to now. In [7], the authors propose a stochastic model, but the underlying assumptions were not adequately confirmed [6] and the model can therefore not be considered as valid. In [8], the authors present a simple model of the TRNG based on coherent sampling. Unfortunately, implementation of the generator is not practical in field programmable gate arrays (FPGA), because it requires topology optimization for each device individually.

In [9], we showed for the first time that self-timed rings (STR) are a highly suitable source of entropy. Based on these observations, in [10] we proposed the first TRNG principle based on STRs.

This paper presents the new self-timed ring based true random number generator (STRNG). It significantly extends the principle presented in [10] by proposing a stochastic model and a design strategy based on this model enabling unprecedented output speed. The feasibility of the generator in logic devices and the plausibility of the new design strategy is demonstrated on two main reconfigurable logic technologies – Altera and Xilinx FPGAs. Raw binary signals and post-processed TRNG output bit streams generated in selected FPGA devices were evaluated using AIS31 testing methodology including FIPS 140-1 statistical tests and also using the NIST statistical test suite [11].

Our contribution: 1) We propose a TRNG principle that enables adjustment of the sensitivity of the entropy extractor to jitter size; 2) we propose a stochastic model of the generator for estimation of entropy per output bit; 3) we propose a TRNG design that makes it possible to manage speed, area, and security according to needs.

The paper is organized as follows: in Section 2 we present the STRNG principle and its design. In Section 3 we describe the stochastic model of the generator and its use for entropy estimation in realistic conditions. In Section 4 we evaluate the feasibility of the STRNG in FPGAs and illustrate it with two implementa-

tions: one in Altera Cyclone III and the second one in Xilinx Virtex 5. In Section 5 we draw some conclusions.

2 Self-timed Ring Based TRNG

Self-timed rings (STR) are oscillators that can provide events which are evenly spaced in time and distributed over half an oscillation period of one ring stage. The time interval between successive events can be set as short as needed and the jitter of each event is mostly composed of the local Gaussian jitter resulting from the ring stage that the event is crossing. In this section, we present a self-timed ring based TRNG (STRNG) based on these features.

2.1 Self-timed Ring Oscillators

STRs use a handshake request and acknowledgment protocol to assure data transfers between adjacent stages. Contrary to inverter ring oscillators, several events can propagate simultaneously in STRs thanks to the asynchronous handshake protocol. On the other hand, STRs exhibit a very specific temporal behavior: for a particular range of numbers of events in relationship with the number of stages, the events lock into a steady state where they propagate with constant spacing, known as the evenly-spaced oscillation mode of an STR. The TRNG proposed in this section exploits two features of the STR:

- If the number of events and the number of STR stages are co-prime, the STR exhibits as many equidistant phases as its number of stages. Its phase resolution can be expressed as follows:

$$\triangle\varphi = \frac{T}{2L}, \tag{1}$$

  where L is the number of STR stages, and T its oscillation period (T can be tuned by the ratio N/L, where N is the number of events). This phase resolution can be set as finely as needed.
- The jitter that appears at the output of each STR stage is mostly composed of the random jitter that originates from the local noise sources of the stage concerned.

Appendix A presents STRs, their architecture, and their temporal behavior in the amount of detail needed to understand the rest of this paper.

2.2 STRNG Principle

Left part of Fig. 1 shows the architecture of the STRNG. If an L-stage STR is initialized with N events and N and L are co-prime, the STR delivers L jittery signals $(C_i)_{1\leq i\leq L}$ spread evenly around the ring and that have the same period T. These signals have a constant mean phase difference $\triangle\varphi = T/2L$. A reference

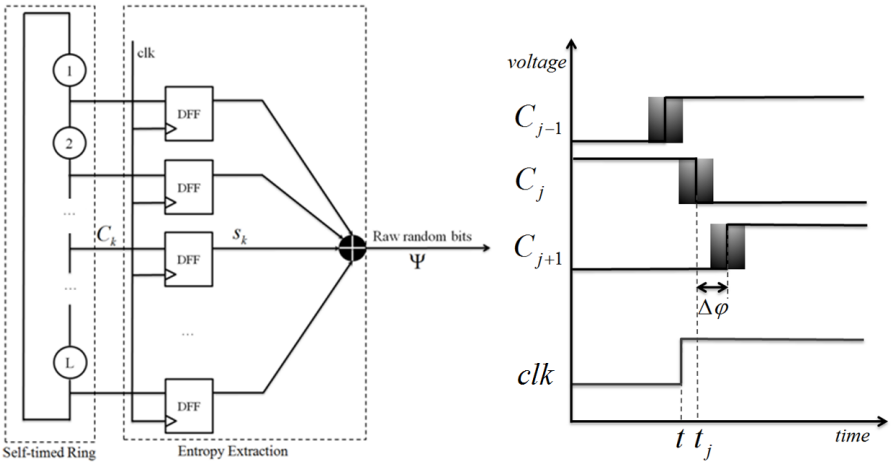


Fig. 1. STRNG core architecture and entropy extraction principle

clock signal *clk* is used for sampling outputs of ring stages using flip-flops. The signals obtained $(s_i)_{1 \leq i \leq L}$ are then combined using a XOR function to obtain the STRNG output $\psi = s_1 \oplus s_2 \oplus ... \oplus s_L$.

Right part of Fig. 1 illustrates the entropy extraction principle. The STR output signals are re-indexed according to their mean arrival time (C_i and C_{i-1} are not adjacent stages). Since each signal C_i is sampled using the same reference clock *clk*, for any sampling instant t, there exists j such that $|t - t_j| \leq \frac{\triangle\varphi}{2}$, where t_j is the switching time of the signal C_j. If the jittery interval around the mean signal phase is longer than the phase difference between two signals $\triangle\varphi$, the signal C_j is sampled in its jittery time interval. The resulting sample s_j then has a random value, and hence the output of the XOR gate is also random. The entropy of the corresponding bit of the STRNG output (signal ψ) is at least equal to the entropy of the sample s_j. The higher the jitter magnitude and the lower the phase difference $\triangle\varphi$, the higher the entropy of the sample s_j and the higher the entropy at the output of the TRNG. If we denote H the Shannon entropy, then:

$$H(\psi) \geq H(s_j) \tag{2}$$

Although the theoretical concept described here does not require a jittery sampling clock, in practical designs, the jitter of the sampling clock enhances the entropy at the output of the TRNG. However, we do not take this jitter into account while setting up the design (i.e. choosing the phase resolution of the STR with respect to its jitter magnitude). In this way, no assumption or constraint on the sampling clock needs to be made (worst case scenario).

2.3 Comparison with the Inverter Ring Oscillator Approach

The entropy extraction in this design is similar to the one used in [7]. But, due to the use of an STR, two major aspects of the behavior of the STRNG differ significantly.

In [7], several inverter ring oscillators are used (each ring providing one periodic signal), but their mutual phases are not controlled (they are supposed to be independent). The setup of the design relies on a probabilistic assumption: if enough ring oscillators are used, the mean elapsed time between successive events is likely to be short enough to enable each sample to happen in a jittery interval around one event. A probabilistic model based on the coupon collector's problem is used to estimate the number of oscillators needed. Conversely, the STR (which provides as many periodic signals as needed) allows a precise setup of the time elapsed between successive events using Eq. (1).

The signals resulting from the STR outputs are synchronized and their mutual position does not change over time. In contrast, the ring oscillator output signals from [7] drift in time and generate pseudo-randomness. This behavior was confirmed by simulations: sequences generated by combining signals from the outputs of 18 ideal inverters (without jitter) oscillating at slightly different frequencies, passed NIST statistical tests ([12]).

3 Stochastic Model of the STRNG

In the next section, we propose a simple stochastic model to estimate entropy per output bit of the STRNG. The objective is to provide a lower bound of entropy per bit as a function of the ring characteristics: number of stages, oscillation period, and jitter size.

3.1 Definitions and Assumptions

The model assumes the presence of a Gaussian random jitter component at the output of each STR stage. This jitter component is caused by an unavoidable thermal noise (a white noise) generated independently in each STR stage. The main practical issue is to correctly measure its magnitude independently from additional noise components. For the sake of simplicity, we suppose that the sampling clock is an ideal jitter-free clock. The idea is to estimate the entropy resulting only from the STR, and to derive its lower bound without any assumption concerning the sampling clock. The model is based on the following observations:

- The STR output signals $(C_i)_{1 \leq i \leq L}$ provide L jittery events, whose mean time values (denoted $(tm_i)_{1 \leq i \leq L}$) are evenly distributed over half an oscillation period. The STR output signals are re-indexed according to the mean time values of their events ($tm_1 \leq tm_2 \leq ... \leq tm_L$). We denote $\triangle\varphi$ the mean time interval between two successive events (which corresponds to the STR phase resolution described by Eq. (1)): $tm_{i-1} - tm_i = \triangle\varphi$

– The effective timing of events are modeled as Gaussian random variables whose mean values are determined by the phase resolution of the STR, and for which the standard deviation corresponds to the standard deviation of the propagation delay of one ring stage. In the following, we refer to this standard deviation simply as jitter magnitude, denoted σ.
– Each signal C_i is sampled at the same time t, the resulting samples $(s_i)_{1\leq i\leq L}$ are combined with a XOR function and ψ is the resulting combined signal.

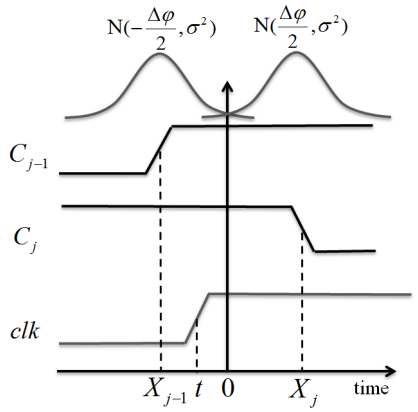


Fig. 2. Modeling of the entropy extraction

Figure 2 shows the modeling of the entropy extraction. For a given sampling time t, two successive events result from signals C_{j-1} and C_j such that $tm_j \leq t \leq tm_{j-1}$. We position the time origin in the middle of the mean time interval between the events (so that $tm_j - tm_j = 0$). This, added to the remarks above, leads to the following definitions:

– The effective time value of the event resulting from signal C_j is a random variable X_j described by a normal law whose mean value is $\frac{\triangle\varphi}{2}$, and whose variance is σ^2. We denote it: $X_j = \mathcal{N}(\frac{\triangle\varphi}{2}, \sigma^2)$
– The effective time value of the event resulting from signal C_{j-1} is a random variable X_{j-1} described by a normal law whose mean value is $-\frac{\triangle\varphi}{2}$, and whose variance is σ^2. We denote it: $X_{j-1} = \mathcal{N}(-\frac{\triangle\varphi}{2}, \sigma^2)$

Note that X_j and $X_j - 1$ are independent random variables because they are related to two different events at distant ring stages. Signal ψ can be decomposed into the sum of $\omega = s_j \oplus s_{j-1}$ and $\mu = \oplus(s_i)_{i\neq j, i\neq j-1}$. We denote $H(\psi)$ the Shannon entropy function of an output bit of the signal ψ (associated with the sampling instant t). It should be noted that $H(\psi) \geq H(\omega)$ because $(s_i)_{1\leq i\leq L}$ are independent samples. This means that we can derive a lower bound of entropy per output bit of ψ by computing the Shannon entropy function of the output bits of ω. In practice, our previous investigations showed that $H(\mu)$ can be

safely neglected unless $\triangle\varphi \ll \sigma$. In that case ($\triangle\varphi \ll \sigma$), μ yields some entropy, but $H(\omega) \simeq 1$ so that $H(\psi) \simeq 1$. Therefore, in the following, we assume that $H(\mu) \simeq 0$ and we denote u the value of the output bit of μ associated with the sampling moment t (u being '1' or '0', but not random). These remarks can be summarized in the following equation:

$$\psi = \omega \oplus \mu \quad and \quad H(\psi) \simeq H(\omega), \tag{3}$$

where $H(\omega)$ is a function of the realizations of random variables X_j and X_{j-1}, described by the following normal laws:

$$X_j = \mathcal{N}(\frac{\triangle\varphi}{2}, \sigma^2) \;\; and \;\; X_{j-1} = \mathcal{N}(-\frac{\triangle\varphi}{2}, \sigma^2) \tag{4}$$

3.2 Binary Probability Computation

First, for a fixed sampling time t, we compute the probability that the output bit value of ψ is equal to u, which we denote $P(u)$. This probability is determined by the realizations of the random variables X_{j-1} and X_j. Table 1 gives the value of ω and ψ as functions of the realizations of X_{j-1} and X_j. $\bar{u}$ is the complementary value of u.

$X_{j-1} \leq t$	$X_j \leq t$	ω	ψ
false	false	'1'	$\bar{u}$
false	true	'0'	u
true	false	'0'	u
true	true	'1'	$\bar{u}$

Table 1. Values of ω and ψ as functions of the realizations of X_{j-1} and X_j and the sampling time t

We denote $p = P(X_j \leq t)$ the probability that $X_j \leq t$, and $p' = P(X_{j-1} \leq t)$ the probability that $X_{j-1} \leq t$. According to Tab. 1, the probability of obtaining a value u in the signal ψ, which we denote $P(u)$, is:

$$P(u) = p + p' - 2pp' \tag{5}$$

The cumulative distribution function (Φ) of the standard normal distribution $\mathcal{N}(0,1)$ describes the probability that the associated random variable falls in the interval $[-\infty, x]$. It is defined as follows:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-t^2}{2}} dt \quad , \; x \in \mathbb{R}$$

From Eq. (4) and from the above definition, we deduce p and p' as functions of t, σ and $\triangle\varphi$: $p = \Phi(\frac{t - \frac{\triangle\varphi}{2}}{\sigma})$ and $p' = \Phi(\frac{t + \frac{\triangle\varphi}{2}}{\sigma})$

Finally, using Eq. (1) and Eq. (5), we express the probability that the output bit value of ψ is equal to u ($P(u)$) with respect to the jitter magnitude (σ), the oscillation period (T), the number of ring stages (L) and the sampling time (t) as follows:

$$P(u) = \Phi(\frac{t - \frac{T}{4L}}{\sigma}) + \Phi(\frac{t + \frac{T}{4L}}{\sigma}) - 2\Phi(\frac{t - \frac{T}{4L}}{\sigma})\Phi(\frac{t + \frac{T}{4L}}{\sigma}) \tag{6}$$

3.3 Lower Bound of Entropy per Output Bit

The Shannon entropy of an output bit of signal ψ, associated with the sampling instant t, is defined as follows:

$$H(\psi) = -P(u)log_2(P(u)) - (1 - P(u))log_2(1 - P(u)), \tag{7}$$

$P(u)$ can be computed using Eq. (6). Therefore, $H(\psi)$ is a function of t, σ, T and L. In the left part of Fig. 3, we plotted $H(\psi)$ as a function of time for $\triangle\varphi$ equal to 10 time units, and for different values of the jitter magnitude σ. As can be seen in these graphs, entropy is maximum when sampling happens at the edges of the signals ($t = \frac{\triangle\varphi}{2}$ and $t = \frac{-\triangle\varphi}{2}$). Conversely, entropy is minimum when sampling happens far from the signal edges ($t = 0$). On the other hand, the higher the jitter magnitude σ, the higher the lower bound of entropy at the output of the TRNG (dotted curves in Fig. 3).

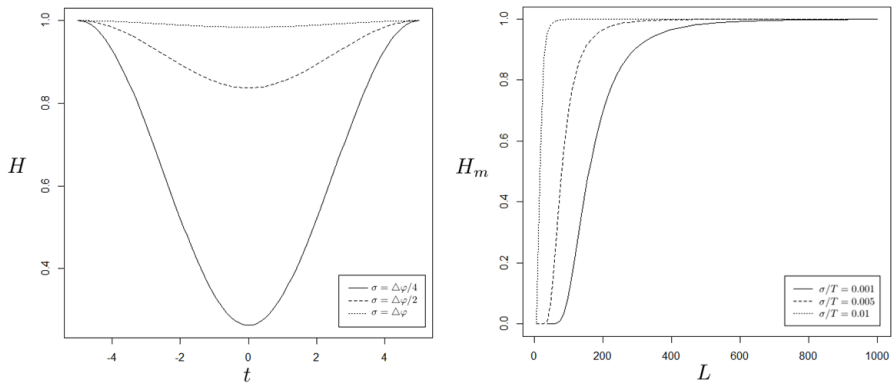


Fig. 3. Entropy of a sampled bit as a function of time and lower entropy bound per output bit with respect to the number of stages

The lower bound of entropy per output bit is obtained by replacing $t = 0$ in the previous equations. From Eq. (6), and knowing that $\phi(-x) = 1 - \phi(x)$ $(x \in \mathbb{R})$, we express $P(u)_{t=0}$ as follows:

$$P(u)_{t=0} = 1 - 2\phi(\frac{T}{4L\sigma}) + 2(\phi(\frac{T}{4L\sigma}))^2 \tag{8}$$

Denoting H_m this lower bound of entropy per output bit, it can be expressed with respect to $P(u)_{t=0}$:

$$H_m = -P(u)_{t=0}log_2(P(u)_{t=0}) - (1 - P(u)_{t=0})log_2(1 - P(u)_{t=0}) \tag{9}$$

H_m is a function of the jitter magnitude σ, the number of STR stages L and their oscillation period T. In the right part of Fig. 3, we plotted H_m as a function of

L for different values of σ/T. We assume that the frequency is maintained when the number of stages is increased by judiciously selecting the number of events (that are still co-prime with the number of stages). As expected, H_m increases with the number of ring stages. As a consequence of this feature, the sensitivity of the entropy extractor can be tuned to jitter size, by simply adjusting the number of STR stages. Moreover, the STRNG can be exploited with optimal entropy ($H_m \geq 0.99$) if the selected number of STR stages is high enough.

3.4 Practical Use of the Model

The purpose of this model is to help designers select the number of STR stages required to achieve a targeted entropy per output bit of the STRNG. This setup requires measuring the STR oscillation period and its jitter magnitude. Using these measurements, designers can plot the entropy curve (similar to the curve in the right part of Fig. 3) and select the number of stages needed to achieve a targeted lower bound of entropy per output bit. The jitter measurement is critical considering its low magnitude in self-timed rings (a standard deviation of few picoseconds); consequently a few precautions need to be taken, and these are discussed in [10].

Fine tuning involving a trade-off between the STRNG size (number of STR stages) and its throughput can be achieved by compressing the output data using a parity filter. An $n^{th}-$order parity filter combines n successive input bits into one output bit using a XOR function, which enhances the entropy per output bit, but reduces the throughput by n. The main advantage of the parity filter is that combined with the proposed stochastic model, it enables simple entropy per bit correction. Supposing that the input bits are independent, $P(u)$ being the input bit probability (u refers to '1' or '0'), the output bit probability $P_{pf}(u)$ is expressed as follows [17]:

$$P_{pf}(u) = 0.5 - 2^{n-1}(P(u) - 0.5)^n \tag{10}$$

Note that the higher the n, the more closely $P_{pf}(u)$ approaches 0.5. Using Eq. (10), designers can recompute the lower bound of entropy by replacing $P(u)$ by $P_{pf}(u)$ in Eq. (9). A trade-off between size and speed can be chosen depending on specific applications and security requirements, by judiciously selecting the filter order n and the number of ring stages L. Throughput loss is mitigated by the fact that no assumption has been made on the sampling clock: its frequency should be as high as permitted by the selected technology. Finally, Appendix B presents a few mainly design-related conditions that should be satisfied in order to guarantee the validity of this stochastic model.

4 Characterization and Evaluation in Altera and Xilinx FPGAs

In this section, we present STRNG designs implemented in Altera Cyclone III and Xilinx Virtex 5 FPGAs. We selected four STR configurations, measured

their oscillation period and jitter magnitude, and computed the lower bound of entropy using the proposed model for each STRNG configuration. Then we evaluated bit sequences acquired at bit-rates up to 400 Mbit/s using AIS31 and NIST SP 800-22 statistical test suites.

4.1 STRNG design

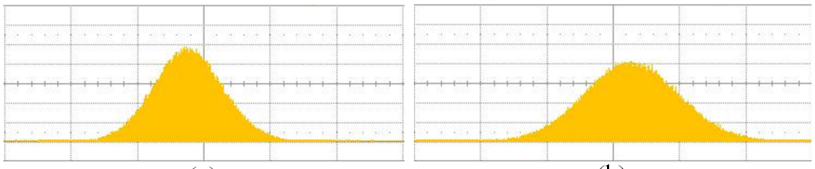
We implemented each STR stage in one look-up-table (LUT) in both Altera Cyclone III and Xilinx Virtex 5. In each LUT, at least four inputs are required: two inputs are used for the stage forward and reverse inputs, one input is used to initialize the stage (SET or RESET), and one input serves as the feedback loop to maintain the state value. The number of events is defined by the the initial values of the STR stages. Both devices feature hard-wired connexions between the LUTs and adjacent flip-flops that we used to connect each stage with its corresponding flip-flop. Ring stages were placed so that the delays between adjacent stages were identical, or at least similar (ring topology). To achieve high working frequencies, we selected ripple architecture for the XOR tree (registers are used between each XOR row). The sampling clock was generated by multiplying an external quartz frequency using the phase-locked loops (PLL) embedded in the selected devices. Sequences were acquired via a USB transfer protocol at 400 Mb/s. For evaluation purposes, we implemented a generic software n^{th}-order parity filter that can be applied to the acquired sequences.

4.2 Characterization of the Entropy Source

We measured the STR frequency and jitter using a wide band digital oscilloscope (LeCroy Wavepro 735 ZI). We used the low-voltage differential signaling (LVDS) outputs of the device and an active differential probe with a 4 GHz bandwidth. We measured the highest frequencies when the number of events was around half the number of stages. Figure 4 shows the period distribution of a 127-stage self-timed ring with 64 events in both Altera Cyclone III and Xilinx Virtex 5. The observed period distribution has a Gaussian shape with a standard deviation of a few picoseconds in both devices. The average jitter magnitude of an STR stage was obtained following the method presented in [9]. Its value was around 2 ps for Cyclone III and 2.5 ps for Virtex 5. This value does not vary with the number of STR stages. For each STR configuration, we measured the oscillation period (T), and then computed the phase resolution ($\triangle\varphi$) using Eq. (1), the lower bound of entropy per output bit (H_m) using Eq. (9), and the minimum filter order (n_{min}) such that $H_m \geq 0.99$ using Eq. (9) and Eq. (10). Results, presented in Tab. 2, are used as a reference for comparison with the statistical evaluation of sequences acquired from different STR configurations.

4.3 Evaluation

For each STR configuration, and each device, we acquired a few Gbytes of raw data from the STRNG output at 400 Mbit/s. We separated the design from



(a) (b)

Fig. 4. Period distribution histogram of a 127-stage STR with 64 events in (a) Altera Cyclone III (b) Xilinx Virtex 5 (scales are 5 *ps* per horizontal division and 100 *kilo sample* per vertical division)

Device	STR		Measurements		Model		Raw data		Compressed data	
	L	N	T	$\triangle\varphi$	H_m	n_{min}	T1-T4	T5-T8	$n_{p_{min}}$	Throughput
Cyclone III	63	32	2.44 ns	19.3 ps	0	-	0%	0/4	7	57 Mbit/s
	127	64	3.11 ns	12.2 ps	0.02	483	0%	0/4	4	100 Mbit/s
	255	128	2.93 ns	5.7 ps	0.58	7	45%	1/4	2	200 Mbit/s
	511	256	3.31 ns	3.2 ps	0.91	2	99%	3/4	2	200 Mbit/s
Virtex 5	63	32	2.82 ns	21.4 ps	0	-	0 %	0/4	8	50 Mbit/s
	127	64	2.83 ns	11.8 ps	0.13	60	10 %	1/4	3	133 Mbit/s
	255	128	2.45 ns	5.5 ps	0.78	4	58%	2/4	2	200 Mbit/s
	511	256	2.87 ns	2.9 ps	0.97	2	61%	3/4	2	200 Mbit/s

Table 2. Oscillation period (T), phase resolution ($\triangle\varphi$), lower entropy bound (H_m), minimum filter order to achieve 0.99 (n_{min}), T1-T4 test passing rates, T5-T8 results, minimum filter order needed to pass tests T1-T8 ($n_{p_{min}}$) and effective throughput for different STR configurations in Altera Cyclone III and Xilinx Virtex 5

surrounding logic such as the communication interface. The generated random data were transfered using LVDS outputs to an acquisition card with sufficient memory. We evaluated acquired data using the AIS31 statistical test suite. Note that tests T1 to T4 correspond to four FIPS 140-1 tests (poker, monobit, runs and long runs). For each configuration, we evaluated 1000 sequences of 20000 bits using T1 to T4 tests. Passing rates are used for qualitative evaluation, they are listed in the column T1-T4 of Tab. 2. We applied T5 to T8 tests on a 1 Mbyte sequence of raw data (column T5-T8 of Tab. 2). Then, for each of these configurations, we used a parity filter and tuned the compression rate so that the sequences passed all the tests (100% T1-T4 passing rate, and successful run of T5-T8). Column $n_{p_{min}}$ indicates the minimum compression rate we had to use to pass all the tests. The throughput column lists the effective bit-rate associated with the compression rate $n_{p_{min}}$.

According to AIS31 recommendations, raw data from the TRNG output, or at least data at the output of the arithmetic post-processing should pass T5 to T8. In Tab. 2, the 511-stage configurations (that yield $\simeq 0.9$ minimum entropy per output bit) passed all these tests except T8 which is the entropy test. Using the model, we computed that we should use a compression rate of 2

in order to obtain sufficient entropy per output bit ($H_m \geq 0.99$). As expected, using this compression rate, data passed all AIS31 tests. It should be noted that $n_{min} \geq n_{p_{min}}$ for all the configurations tested: the compression rates needed in practice are lower that those computed using the model. It should also be noted that some configurations provide practical security (e.g. 127-stage STR with a compression rate of 4 passes all the tests), but do not guarantee theoretical security (the entropy assessment does not meet the requirements).

Finally, we applied a complete run of the NIST test suite on 1000 successive sequences of 10^6 bits with a 0.01 confidence level, acquired from the 511-stage STR configurations. Data obtained from the STRNG passed all the NIST tests in Cyclone III with a compression rate of 3. The effective throughput was 133 Mbit/s. Data acquired from Virtex 5 passed the NIST tests with a compression rate of 4 (giving 100 Mbit/s).

5 Conclusions

In this paper, we presented a true random generator (TRNG) and its stochastic model. This generator exploits the jitter of multiple clock signals extracted from a self-timed ring (STR) to generate random bit sequences at a very high bit rate. The technique takes advantage of specific STR features that allow the time interval between successive events to be set as short as needed, even in the order of picoseconds. This time interval can be set in concordance with the clock jitter magnitude in order to extract the desired level of entropy in the generated bit stream. The proposed stochastic model will help designers compute a lower entropy bound as a function of the STR characteristics, i.e. the number of stages, the oscillation period, and the jitter magnitude. With the entropy assessment they obtained, designers can set the compression rate of the arithmetic post-processing block so as to reach the required security level determined by the entropy per output bit. Finally, we also describe a complete and systematic method for designing such a TRNG. The approach was validated using two different FPGA families to demonstrate the feasibility and the simplicity of the STRNG implementation on standard technologies such as Altera and Xilinx FPGAs. STRNGs can provide high quality random bit sequences that pass AIS31 statistical tests at rates as high as 200 Mbit/s, and NIST statistical tests at rates as high as 100 Mbit/s. Future works will include implementation of the STRNG in an application specific integrated circuit (ASIC), a proposal for design specific embedded tests, and if possible, embedded measurements of the entropy source.

Acknowledgment

We wish to thank Nathalie Bochard for her help in the experimental measurements, and Florent Bernard for his helpful advice. This research is funded by the Rhone-Alpes region (France).

References

1. Fischer, V.: A closer look at security in TRNGs design. In: Constructive Side-Channel Analysis and Secure Design (COSADE 2012). LNCS vol. 7275, pp. 167–182. Springer-Verlag Berlin Heidelberg (2012)
2. Killmann, W., Schindler, W.: A proposal for Functionality classes for random number generators, version 2.0. Bundesamt fur Sicherheit in der Informationstechnik (BSI). Bonn Sept. 2011. url: https://www.bsi.bund.de/EN/Home/home_node.htm
3. Tkacik, T.: A Hardware Random Number Generator. In: Cryptographic Hardware and Embedded Systems – CHES 2002. LNCS, vol. 2523, pp. 450–453. Redwood Shores, CA, USA, Springer Verlag (2003)
4. Majzoobi, M., Koushanfar, F., Devadas, S.: FPGA-Based True Random Num-ber Generation Using Circuit Metastability with Adaptive Feedback Control. In: Preneel, B., Takagi, T. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2011. LNCS, vol. 6917, pp. 17–32. Nara, Japan, Springer Verlag (2011)
5. Fischer, V., Drutarovsky, M.: True Random Number Generator Embedded in Reconfigurable Hardware. In: Cryptographic Hardware and Embedded Systems – CHES 2002. LNCS, vol. 2523, pp. 415–430. Redwood Shores, CA, USA, Springer Verlag (2002)
6. Dichtl, M., Golic, J.D.: High-Speed True Random Number Generation with Logic Gates Only. In: Cryptographic Hardware and Embedded Systems – CHES 2007. LNCS, vol. 4727, pp. 45–61. Vienna, Austria, Springer Verlag (2007)
7. Sunar, B., Martin, W.J., Stinson, D.R.: A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. In: IEEE Transactions on Computers. Vol. 58, pp. 109–119 (2007)
8. Bernard, F., Fischer, V., Valtchanov, B.: Mathematical Model of Physical RNGs Based on Coherent Sampling. Tatra Mt. Math. Publ. 45, 1–14 (2010)
9. Cherkaoui, A., Fischer, V., Aubert, A., Fesquet, L.: Comparison of Self-timed and Inverter Ring Oscillators as Entropy Sources in FPGAs. In: Design, Automation and Test in Europe, DATE 2012. Proceedings of DATE 2012, pp. 1325–1330. Dresden, Germany (2012)
10. Cherkaoui, A., Fischer, V., Aubert, A., Fesquet, L.: A Self-timed Ring Based True Random Number Generator. In: International symposium on advanced research in asynchronous circuits and systems – ASYNC 2013. Proceedings of ASYNC 2013, pp. 99–106. Santa Monica, California, USA (2013)
11. A statistical test suite for random and pseudorandom number generators for cryptographic applications. In: NIST Special Publication (SP) 800-22 rev. 1 (2008). Available at http://csrc.nist.gov/CryptoToolKit/tkrng.html
12. Bochard, N., Bernard, F., Fischer, V., Valtchanov, B.: True-Randomness and Pseudo-Randomness in Ring Oscillator-Based True Random Number Generators. In: International Journal of Reconfigurable Computing. Vol. 2010, article ID 879281 (2010)
13. Winstanley, A., Greenstreet, M. R.: Temporal Properties of Self-Timed Rings. In: 11th Advanced Research Working Conference on Correct Hardware Design and Verification Methods, CHARM01. London, UK, Springer-Verlag, pp. 140–154 (2001)
14. Fairbanks, S.: High Precision Timing Using Self-timed Circuits. In: Technical report no. UCAM-CL-TR-738, University of Cambridge, Computer Laboratory (2009). Available at http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-738.pdf
15. Hamon, J., Fesquet, L., Miscopein, B., Renaudin, M.: High-Level Time-Accurate Model for the Design of Self-Timed Ring Oscillators. In: International symposium on

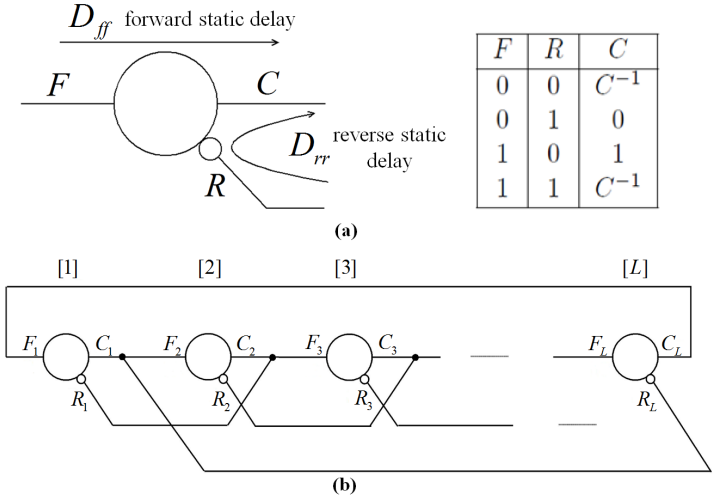
advanced research in asynchronous circuits and systems – ASYNC 2008. Proceedings of ASYNC 2008, pp. 29–38 (2008)
16. Sutherland, I. E.: Micropipelines. In: Communications of the ACM (Association of Computing Machinery). Vol/Issue: 32/6, pp. 720–738 (1989)
17. Davies, R. B.: Exclusive OR (XOR) and hardware random number generators (2002). Available at http://www.robertnz.net/pdf/xor2.pdf
18. Elissati, O., Yahya, E., Rieubon, S., Fesquet, L.: A novel high-speed multi-phase oscillator using self-timed rings. In: International conference of Microelectronics. ICM 2010, pp. 204–207 (2010)

Appendix

A Self-timed Rings

Self-timed rings (STR) are oscillators in which several events can propagate simultaneously without colliding thanks to a handshake request and acknowledgment protocol. They are ripple first-in-first-out memories (FIFOs) that have been closed to form a ring. These FIFOs use an asynchronous handshaking protocol to transfer data between adjacent stages. When closed, they retain the handshaking mechanism that ensures data ordering, but exhibit a very specific temporal behavior: for a particular range of numbers of events in relationship with the number of stages, the events lock in a steady state where they propagate with constant spacing, known as the evenly-spaced oscillation mode of an STR. A detailed description of STRs behavior can be found in [13], [14] and [15].

Architecture The architecture of an STR is depicted in Fig. 5. It corresponds to an asynchronous micropipeline, proposed by Sutherland in [16], that has been closed to form a ring of L stages. Each stage is composed of a Muller gate and an inverter. In Fig.5, D_{ff} and D_{rr} are the forward and reverse static propagation delays of one ring stage associated with inputs F and R.



F	R	C
0	0	C^{-1}
0	1	0
1	0	1
1	1	C^{-1}

Fig. 5. (a) Structure of one STR stage and its truth table - (b) STR architecture

Behavior The micropipeline stages communicate using a two-phase handshake protocol as described in [16]. Each request and acknowledgment signifies an event transfer between interconnected stages. Contrary to inverter ring oscillators, several events can propagate without colliding thanks to the handshake protocol. The ring is initialized with N events that start propagating during a transient state. They eventually end up in a steady state where they arrange themselves

in one of two ways: either they form a cluster that propagates around the ring (burst oscillation mode), or they spread out around the ring and propagate with constant spacing (evenly-spaced oscillation mode). Both oscillation modes are stable and depend on the static parameters of the STR (e.g. the initial value of individual stages and the ratio of forward and reverse propagation delay of one stage). In the evenly-spaced oscillation mode, the event propagation is self-timed: inherent analog mechanisms regulate the time that elapses between successive events. Figure 6 illustrates the evenly-spaced propagation of 2 events in a 5-stage STR.

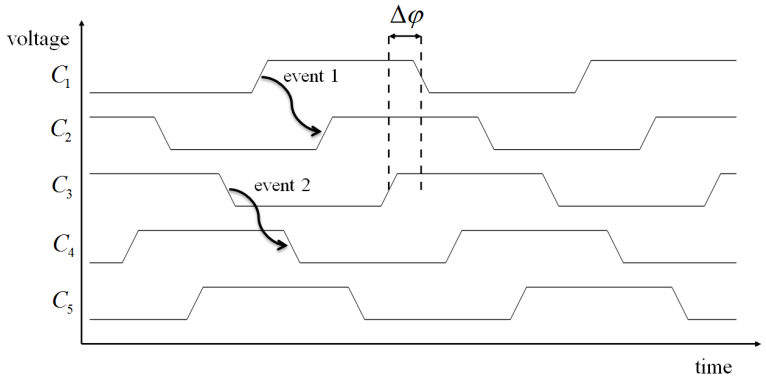


Fig. 6. Chronogram of the evenly-spaced propagation of 2 events in a 5-stage STR

Evenly-spaced Mode Locking Phenomenon The propagation delay of a Muller gate is a function of the separation time between its two inputs. The shorter the separation time, the longer the propagation delay. This phenomenon is called the analog Charlie effect. In the STR context, the Charlie effect causes two close events to push away from each other (in time) due to the increased delay experienced by a ring stage when driven by a request and acknowledge signals with a short separation time. When a large number of events is constrained in a short structure, this effect is retroactive: each event pushes away from its neighbors until they spread out evenly around the ring. The final state of the STR (oscillation period, phase distribution) does not depend on the initial separation times between the events, but rather on the ratio between the number of events and the number of stages (N/L).

Frequency Curve The frequency of an STR in the evenly-spaced regime is a function of its occupancy. The frequency increases with the number of events N (which propagate along the request paths), then starts dropping when the number of free stages is lower than the number of events to process. In this case, the apparent number of propagating events is $L - N$ and the events propagate across the paths of the acknowledge signals. The number of events achieving the

maximum frequency (N_0) is described by the following equation ([15]):

$$\frac{N_0}{L - N_0} \simeq \frac{D_{ff}}{D_{rr}} \tag{11}$$

Generation of Multiphase Signals Contrary to inverter ring oscillators, STRs allow phase resolutions, which are fractions of the propagation delay of a single stage. An event propagation in an STR causes a 90^o phase shift of the oscillating signal. If N events are confined in L stages and spread evenly around the ring, the phase shift between two stages separated by n stages is [14]:

$$\varphi_n = n \times \frac{N}{L} \times 90^o \tag{12}$$

Therefore, if the number of stages is a multiple of the number of events, some stages may exhibit the same absolute phase. But if the number of events and the number of stages are co-prime, the STR exhibits as many different equidistant phases as the number of stages. If T is the oscillation period, the phase resolution can be expressed as follows:

$$\triangle\varphi = \frac{T}{2L} \tag{13}$$

The oscillation period of an STR is a function of its occupancy rather than of the number of its stages. This means that it is possible to increase the number of ring stages (L) while keeping a constant frequency. Consequently, the phase resolution of an STR can theoretically be set as finely as needed. Elissati *et al.* demonstrated the efficiency of the method in [18] by implementing several designs and obtaining phase resolutions in the order of picoseconds.

Jitter Characteristics Each event that crosses a stage of the STR experiences a timing variation due to the local noise sources of the stage. However, the propagation of these timing variations from one stage to another is very limited as the the time that elapses between successive events is controlled by the locking mechanisms explained above. Furthermore, global noise sources (e.g. power supply noise) do not strongly affect the elapsed time between successive events as they have the same impact on each event. Authors in [9] analyze and measure the jitter in STRs implemented in FPGAs. Experimental measurements confirmed a Gaussian distribution of the period with a standard deviation of the same order of magnitude as the propagation delay of one single ring stage. This suggests that the jitter that appears at the output of each STR stage is mostly composed of the random jitter that originates from the local noise sources of the stage.

B Theoretical and Practical Limits of the Stochastic Model Presented in Section 3

This section details a few conditions related to the entropy extraction for the validity of the stochastic model presented in section 3.

Maximum Theoretical Throughput The minimum time interval between two successive samples should be higher than $2\triangle\varphi$ in order to avoid sampling the same jitter realization twice:

$$F_{clk} \leq \frac{1}{2\triangle\varphi} \tag{14}$$

Phase Distribution at Inputs of Flip-flops While the time intervals between the events are self-controlled in the micropipeline, their distribution at inputs of flip-flops depends on the delays between the micropipeline outputs and the corresponding flip-flops. These delays should be identical in order to maintain a uniform phase distribution at the flip-flop inputs. Noting these delays $(D_i)_{1\leq i\leq L}$, we derived the following equation that should be checked to guarantee the validity of the model:

$$Max(|D_i - D_j|)_{1\leq i,j\leq L} \leq \triangle\varphi \tag{15}$$

Clock Skew The assumption that for every sampling instant there exists j such that $|t - t_j| \leq \frac{\triangle\varphi}{2}$ requires that all the effective sampling times of the flip-flops (depending on the clock skew) are constrained in a $\triangle\varphi$ interval. If we denote D_{skew_i} the skew associated with the signal clock feeding the flip-flop i, we derive the following condition for the model to hold:

$$Max(|D_{skew_i} - D_{skew_j}|)_{1\leq i,j\leq L} \leq \triangle\varphi \tag{16}$$

Dependence between Successive Output Bits and Conditional Entropy Let $(X_i)_{1\leq i\leq n}$ be a sequence of output bits of the STRNG. The model presented in this paper assumes output bits are independent, which is the condition for applying Eq. 10. This assumption is based on our observation that, unlike in most digital oscillators, timing variations between two sampling events are reduced due to the analog effects that control the timings in the STR. In this case, the conditional entropy of an output bit of the STRNG (i.e. the entropy of output bit X_n when the preceding sub-sequence is known) approaches the entropy of this output bit without knowledge of its predecessors:

$$H(X_n) \simeq H(X_n|X_{n-1}, ..., X_1) \tag{17}$$